CLOCKWORKS®

USER'S MANUAL

MICRO SYSTEMS RESEARCH
4099 Maxanne Drive, Kennesaw, GA 30144

CLOCKWORKS®

AN ADVANCED REAL TIME CLOCK
FOR THE APPLE II+, AND IIe

USER'S MANUAL

Written by

Maurice Khano
and
Dav Holle

Revision D

Micro Systems Research
4099 Maxanne Drive
Kennesaw, Georgia 30144
(404) 928-9394

Se

Congratulations on your purchase of a Micro Systems Research CLOCKWORKS Real Time Clock. Due to the excellent care taken in the design and construction of this product, we are confident that you'll enjoy its reliable operation for many years to come.

Micro Systems Research is proud to bring you the highest quality products at affordable prices, excellent service and outstanding customer support.

Constructive criticism is welcomed at all times, especially comments that may lead to improvements in product usefulness and/or manual content or format.

If you have special requirements and would like to discuss your software and/or hardware needs do not hesitate to drop us a note or call us at (404)928-9394.

We'll be happy to help...

Trademarks

Clockworks is a registered trademark of Micro Systems Research.
Apple, Appleworks, and ProDOS are trademarks of Apple Computer Inc.
Thunderclock is a trademark of Thunderware Inc.
Timemaster II H.O. is a trademark of Applied Engineering.
Appleclock is a trademark of Mountain Computer Inc.
UCSD Pascal is a trademark of the regents of the University of California.
MultiScribe is a trademark of Styleware Inc.
MacroWorks is a trademark of Beagle Brothers.

TABLE OF CONTENTS

CHAPTER I
INTRODUCTION

Your new Clockworks Real Time Clock will add a whole new dimension to your Apple computer. The applications are unlimited. Some uses are:

-- Automatic time and date stamping of files on disk.

-- Adding the time and/or date to any printed hard copy.

-- Measurement of time elapsed between two events.

--Recording duration of phone calls to information systems.

-- Data acquisition and logging, plus many more...

Clockworks is a sophisticated real time clock that is capable of delivering the time in many different formats. The powerful built-in firmware allows it to emulate Thunderclock, Appleclock, and Timemaster, but none of these can do what Clockworks does.

Your new Clockworks advanced real-time clock was designed in 1985 using the most current advancements in technology. We are confident that you will find Clockworks a most valued addition to your Apple.

*** IMPORTANT ***

We urge you to make backup copies of the software supplied right now before you proceed with the installation. Any standard copy program will work such as Apple's COPYA program, or Central Point Software's COPY II PLUS. Remember to copy both sides of the disks, store the originals in a safe place, and use the backup disks only.

CHAPTER II
INSTALLATION

To properly install your Clockworks card into your Apple, follow these instructions:

1. **Make sure your Apple is turned off.**

2. Remove the Apple's top cover by gently lifting the back end from both corners.

3. Insert Clockworks into any available slot. (except 0 in Apple II+ and 3 in Apple IIe equipped with 80-column or extended 80-column cards).

4. Replace the Apple's cover.

5. Insert the backup of disk one supplied into your first drive.

6. Turn on the power to your Apple.

You should now have a menu on the screen. Type "1" to run a program which will display several time/date formats on the screen. If the running time appears with the top of the screen showing which slot you've selected then all is well and the installation is complete. To change the time setting, reboot and select the appropriate number from the menu.

CHAPTER III
SETTING THE SWITCHES

A bank of eight switches is located near the top and center of the card. The purpose and use of these switches is as follows:

Switch	Normal setting	ON	OFF
1	ON	Reserved for future expansion (no effect)	Reserved for future expansion (no effect)
2	ON	Enable setting the time	write protect
3	ON	Enable IRQ type interrupts	Disable IRQ type interrupts
4	OFF	Enable NMI type interrupts	Disable NMI type interrupts
5	ON	Enable 1 per minute interrupts	Disable 1 per minute interrupts
6	ON	Enable 1 per hour interrupts	Disable 1 per hour interrupts
7	OFF	Connect CB1 handshake line to aux. port	Disconnect CB1 handshake from auxiliary port
8	OFF	Connect CB2 handshake line to aux. port	Disconnect CB2 handshake from auxiliary port

Most users will not need to change the setting of these switches. The factory setting is compatible with all existing applications software (such as AppleWorks, DB Master, etc...) while more experienced users, programmers, and OEMs can configure the switches to harness additional features that can eliminate the need for other costly interface cards.

Although no accidental writes to the clock have ever been noted, users can absolutely write-protect Clockworks by turning switch #2 to the OFF position. This feature can be used to prevent other users of a system from changing the clock settings.

CHAPTER IV
USING CLOCKWORKS

Your Clockworks can be easily instructed from Applesoft (DOS 3.3 or ProDOS) to supply the date and time at any point in your program. Also most application packages that require or are capable of using a real-time clock are automatically compatible with Clockworks. Some of these are:

- Appleworks (Apple Computer Inc.)
- ReportWorks (Megahaus)
- MegaWorks (Megahaus)
- MacroWorks (Beagle Brothers)
- Jeeves (Rev. 1.1) PBI Software
- Pinpoint (Pinpoint Publishing)
- Pascal Speedup Kit (Stellation Two)
- Micro Telegram (Microcom)
- Visidex (Visicorp)
- Time Manager (Microsoft)
- Networks (Advanced Data Systems)
- Desk Calendar II (Telephone Software Connection)
- Data Dex (Information Unlimited Software)
- Transcend 3 (SSM)
- CPApartner (Software Dimensions)
- PROpartner (software dimensions)
- Softerm (Softronics)
- OS9 (Stellation Two)
- Accounting Plus Super/E (Software Dimensions).
- DB MASTER (Stone Edge).
- Cashier (High Technology).
- Micro-Courier (Microcom).
- Store Manager (High Technology).
- Z-Term, The Professional (Roger Wagner Publishing).

Plus many more...

For Specific instructions on use with the above software please consult the manual included with the package. Many other programs that use the Mountain Appleclock format are also compatible with your Clockworks card.

4.1 TIME AND DATE FORMATS

Clockworks can supply the time and date information in any of eight different formats. The desired mode is easily selected by printing a single character to the slot where Clockworks is installed. The following list shows the mode setting characters and the resulting formats.

EXAMPLE: Monday, June 24, 1985. 2:35 PM and 45 seconds.

```
" "    MO/DD  HH:MI:SS.WYY          APPLECLOCK
       06/24  14:35:45.185          APPLESOFT BASIC

 $     WWW  MMM  DD  HH:MI:SS PM    THUNDERCLOCK
       MON  JUN  24  02:35:45 PM    APPLESOFT BASIC

 &     WWW  MMM  DD  HH:MI:SS       THUNDERCLOCK
       MON  JUN  24  14:35:45       APPLESOFT BASIC

 #     MO,OW,DD,HH,MI,SS            THUNDERCLOCK
       06,01,24,14,35,45            APPLESOFT BASIC

 >     WWW  MMM  DD  HH:MI:SS PM    THUNDERCLOCK
       MON  JUN  24  02:35:45 PM    INTEGER BASIC

 <     WWW  MMM  DD HH:MI:SS        THUNDERCLOCK
       MON  JUN  24 14:35:45        INTEGER BASIC

 :     W  MO/DD/YY HH:MI:SS                 CLOCKWORKS STRING (TIMEMASTER
COMPATIBLE)
       1 06/24/85 14:35:45          APPLESOFT BASIC

 =     OW,MO,DD,YY,HH,MI,SS         CLOCKWORKS NUMERIC
       01,06,24,85,14,35,45         APPLESOFT BASIC
```

As indicated, some of the formats are designed for use with Applesoft while others are used with Integer Basic.

The Applesoft formats containing colons are preceded with a quotation mark this is so that Applesoft will accept the colons instead of printing a syntax error. The formats designed for Integer Basic are preceded with a space character.

4.2 SETTING THE CLOCK

In order to set a new time and/or date, switch #2 must be closed. There is a program called SET CLOCKWORKS on the accompanying diskette. Running this program will automatically find the slot used and allow you to set the clock. Note that in setting the week you'll enter a number ranging from 0 to 6, with 0 representing Sunday, 1 representing Monday and so forth, also you are required to enter the hours in military format (0-23).

4.3 SETTING THE CLOCK FROM WITHIN YOUR PROGRAM

In addition to the "SET CLOCKWORKS" program supplied on disk, there are two SET MODE commands built right into your CLOCKWORKS card. The advantage to using these commands is that the programmer does not have to load a lengthy program into memory in order to set the time. Instead, the clock can be set by simply printing a string of characters to the clock. The string should contain the data of the new clock setting according to one of the following formats:

CLOCKWORKS SET MODE: +W MO DD YY HH MI SS <RETURN>

THUNDERCLOCK SET MODE: !MO W DD HH MI SS <RETURN>

The spaces shown above do not have to be included as part of the string, nor does it matter how many spaces are used to separate the digits. Each occurrence of a digit increments the format counter for the SET MODE used, so ZERO's must be included in two digit values that are less than 10.

The meaning of the format notation is as follows:

W One digit defining the day of the week, where 0=Sunday,1=Monday,...,6=Saturday.

MO Two digits (01-12) defining the month, where 01=January,02=February,...,12=December.

DD Two digits (01-31) defining the date in the month.

YY Two digits (00-99) defining the year.

HH Two digits (00-23) defining the hour in 24 hour format, where 00=Midnight, 01= 1 AM,..., 23= 11 PM.

MI Two digits (00-59) defining the minute.

SS Two digits (00-59) defining the second.

The Thunderclock set mode, activated by the '!' character, has been included for compatibility with Thunderclock. Thunderclock does not handle year information so its set mode does not include the year data. The CLOCKWORKS SET MODE, activated with the '+' character, does take the year into account so we recommend using the CLOCKWORKS SET MODE for all newly written applications.

The following Applesoft program sets the clock to MONDAY APRIL 21, 1986 4:57:23 PM. Your CLOCKWORKS card is assumed to be in slot #2.

```
100 PRINT CHR$(4);"PR#2"
110 PRINT "+1 04 21 86 16 57 23"
```

```
120 PRINT CHR$(4);"PR#0"
```

You do not necessarily have to set all the data in your clock. Any portion from the beginning of the format can be used. For example changing line 110 above to:

```
110 PRINT "+2 05 13"
```

simply changes the date to Tuesday May 13 without affecting the time or year information in any way! The minus (or dash) "-" character if printed within the set string data has a special meaning. It is used to SKIP OVER the data that you do not want changed. Changing line 110 above to the following sets the hour to 9 AM without affecting the date, minutes, or seconds.

```
110 PRINT "+- -- -- -- 09"
```

One application for this function is when changing to or from daylight savings time. Also, if you want to change the day of week and date in the month to MONDAY 12th you would replace line 110 above with the following:

```
110 PRINT "+1 -- 12"
```

The following program allows a new time to be entered and sets the clock according to the data supplied:

```
100 SLOT=4                                    :REM CLOCKWORKS' SLOT
110 TEXT:HOME                                 :REM CLEAR THE SCREEN
120 D$=CHR$(4)                                :REM CONTROL D
130 PRINT "FORMAT: W MO DD YY HH MI SS"       :REM SHOW FORMAT
140 INPUT "SET TO: ";TM$                      :REM COLLECT NEW ENTRY
150 PRINT D$;"PR#";SLOT                       :REM OUTPUT TO CLOCK
160 PRINT "+";TM$                             :REM SET NEW DATE/TIME
170 PRINT D$;"PR#0"                           :REM RETURN OUTPUT TO SCREEN
```

4.4 READING THE CLOCK

Reading the date and time is very easy because all the software drivers necessary are right on the card itself in a 4,096 Byte EPROM, this is 2 to 8 times larger than most other clock/calendar cards. To read the time and/or date, all you do is direct input and output to the clock, issue an input statement and return input and output to the keyboard and screen. The following will do just that.

```
100 D$=CHR$(4)                          :REM CONTROL D
110 SLOT=4                              :REM CLOCKWORKS' SLOT
120 PRINT D$;"IN#";SLOT                 :REM INPUT FROM CLOCK
130 PRINT D$;"PR#";SLOT                 :REM OUTPUT TO CLOCK
140 INPUT ":";TM$                       :REM GET THE TIME
150 PRINT D$;"PR #0"                    :REM OUTPUT BACK TO SCREEN
160 PRINT D$;"IN #0"                    :REM INPUT BACK TO KEYBOARD
```

After your Apple executes the above program the string TM$ will contain the date and time in the

Clockworks String Format (Timemaster compatible). Adding the following two lines will allow you to display the clock on a continuous basis.

```
170 VTAB 10:HTAB 10:PRINT TM$
180 GO TO 120
```

You can also replace the colon in line 140 above with any of the string formats command characters. To use a numeric mode you do things a little differently. Because the parameters are separated by comma's you must use a variable for each. This makes it easy to parse the date and time without resorting to string manipulation functions. To select the Clockworks numeric mode and read the date and time you do the following:

```
100 D$=CHR$(4)
110 SLOT=4
120 PRINT D$;"IN#";SLOT
130 PRINT D$;"PR#";SLOT
140 INPUT "="; DW,MO,YR,DM,HH,MI,SS
150 PRINT D$;"PR#0"
160 PRINT D$;"IN#0"
```

This gives you each piece of information in a separate variable. You also could have parsed the string from our previous example by using the LEFT$,MID$, and RIGHT$ Applesoft commands and then using the VALUE function to obtain a numeric value. Once you have a value for the week or month you can convert it to a word like this:

```
IF MO=1 THEN MO$="JANUARY"
IF MO=2 THEN MO$="FEBRUARY"
etc....
```

Or you can put the names of all the months in a data statement and assign MO$ based on the value in MO. We don't have room to show you that but you get the message. You obviously can do the same for the day of the week or if you want to spell out the numbers, its up to you!

4.5 READING THE CLOCK WHILE IN 80 COLUMN MODE

Other real time clock cards have a problem with the 80 column display when they are being used from Applesoft. The problem creeps in when we try to restore the output to the screen. If we restore output with a PR#3 command to go to 80 columns, (instead of PR#0 for 40 column screen) the screen is erased and any window settings are gone! CLOCKWORKS resolves this by providing you with a special command to read the date & time from within your Applesoft programs without reinitializing your 80 column display (or the current output device). The format is as follows:

```
CALL 49216 + 256 * SLOT ,FM$,TM$
```

Where FM$ is a one character string indicating the read format, and TM$ is the variable name in which the clock data will be placed. For example, the following program reads the time & date in the Thunderclock AM/PM mode:

```
100 CALL 49216+256*SLOT,"%",TM$
110 PRINT TM$
```

Where "SLOT" is the number of the slot used. Or for slot 4 you can use:

```
100 CALL 50240,"%",TM$
```

This command is actually faster and easier to use than the vectored I/O methods (PR#,IN#) described in previous sections of this manual. It also does not produce an awkward looking flashing cursor when the time is being read continuously. This is compatible with 40 column screens as well.

The following program continuously displays the time & date in all the read modes:

```
100 PRINT : PRINT CHR$(4);"PR#3"
110 SLOT=4
120 HOME : PRINT : PRINT
130 PRINT "MODE","CLOCK","LANGUAGE","READ FORMAT"
140 VTAB 6
150 FOR I= 1 TO 8
160 READ MODE$,CLOCK$,LANG$
170 CALL 49216 + 256 * SLOT,MODE$,TIME$
180 PRINT MODE$,CLOCK$,LANG$,TIME$
190 NEXT
200 RESTORE
210 GOTO 140
220 DATA " ","APLCLK","APLSFT","%","THUNCLK","APLSFT"
230 DATA "&","THUNCLK","APLSFT","$","THUNCLK","APLSFT"
240 DATA ">","THUNCLK","INTBAS","<","THUNCLK","INTBAS"
250 DATA ":","TMHO","APLSFT","=","CLKWRKS","APLSFT"
```

4.6 HOW A PROGRAM FINDS CLOCKWORKS

From within your program you can search and locate a Clockworks card in your Apple. The following program does this by searching for specific signature bytes in the firmware. Do not type this program, it is already on your Clockworks Diskette.

```
150 SLOT=0:I=1
160 ADDR=49152+256*I:REM $CN00
170 IF PEEK(ADDR)=8 AND PEEK(ADDR+252)=195 AND PEEK(ADDR+253)=215 THEN SLOT=I: GOTO200
180 I=I+1:IF I>7 THEN 200
190 GO TO 160
200 IF SLOT=0 THEN PRINT "CLOCKWORKS NOT FOUND": END
210 PRINT "YOUR CLOCKWORKS IS IN SLOT ";SLOT
```

The above code may be incorporated into your own programs to locate the clock and set the variable "SLOT" which is usually used later to read the clock data.

4.7 PRODOS TIME AND DATE STAMPING

One of the nice features of Clockworks is its automatic emulation of Thunderclock. This allows ProDOS to automatically time and date stamp your files and programs when they are saved or updated. The ProDOS (or more accurately Basic.system) "CATALOG" command displays the date & time of when the file was originally created and that of when it was last modified. Every time you save or update a file or program, ProDOS calls upon Clockworks to supply the date & time. Appleworks, Reportworks and all other ProDOS based applications can automatically read Clockworks to time & date stamp your files.

4.8 DOS 3.3 TIME AND DATE STAMPING

Unlike ProDOS and Pascal, DOS 3.3 does not normally stamp files saved to disk with the date or time information, but a program is included on your Clockworks DOS 3.3 disk that installs the necessary patches to allow the date and/or time data to be added to all files SAVED, BSAVED, or RENAMED.
The program that automatically installs these patches for you is available on the DOS 3.3 disk supplied and is easily selected from the bootup menu, or can be executed by typing the following from Basic:

```
RUN INSTALL DOS DATE STAMP PATCH  <PRESS RETURN>
```

Once this program is executed, several date/time formats are displayed with an arrow pointing to the first one. This is the recommended format because it provides the year information. To select this format simply press <RETURN>. To select an alternative format use the up/down arrow keys then press <RETURN>.

Another display now appears showing the format you selected with a highlighting bar underneath it

which indicates the portion of the string to be used. Either accept the portion shown by pressing <RETURN> or adjust the field with the following commands:

S	Shortens the field from the left edge
L	Lengthens the field from the left edge
-->	Move field to right
<--	Move field to left

If you want to go back to change your format selection, you can press <ESCAPE>. Press <RETURN> to complete the installation.

Now the DOS in memory has your modifications installed and you may format as many disks as you like with the DOS "INIT" command and then move any files from the software disk you want to use to the newly formatted disk. To directly install the new DOS on your disks you can use a utility program called "THE FILER" available from Central Point Software that will simply replace the disk's DOS with the DOS in memory.

The above described patch is compatible with most unprotected software . Any new files SAVED, BSAVED, or RENAMED will have the date/time appended to the filenames. The INIT command will also add the date/time to the boot program's name.

If you need to access a time/date stamped file while running an unmodified version of DOS you can do so by typing the full name of the file including the date/time with exactly the same number of spaces in between or you can easily patch DOS directly in memory to cause it to ignore the date/time stamp. To do this type:

```
POKE -19965, N          limits compare size in catalog search routine
POKE -22653, N-1        limits compare size in OPEN text file handler
```

Where N is a number between 1 and 30 and indicates how many characters from the beginning of the filename are significant. This number is calculated to be 30 - (length of date/time stamp). If you've used the default format and portion, (ie 10/23/86) with the leading space, then N would be 21.

CHAPTER V
THE ULTRASONIC REMOTE CONTROL OPTION

The ultrasonic remote control option for CLOCKWORKS allows your Apple to transmit control commands to your BSR X-10 command console. This gives your computer the ability to control lights , appliances, and most other devices that operate on standard AC current. The ultrasonic command console is available from several suppliers as the 'BSR X-10 Model UC301' or from SEARS as model X-10-014301. This is the model that operates with the handheld cordless controller.

5.1 INSTALLING THE BSR X-10 INTERFACE

To install the BSR X-10 remote control interface, follow this procedure:

1. **Turn OFF the power to your Apple.**
2. Remove the Apple's cover by lifting the back end from both corners.
3. Unplug your CLOCKWORKS card.
4. Locate the AUXILIARY connector on the card. (This is the gold connector at the top right hand corner of the printed circuit board.
5. Locate pin #1 and pin #16 on the two opposing corners of the connector. (refer to AUXILIARY connector pinout if necessary)
6. Plug one end of the BSR cable on pin 1 and the other on pin 16.
7. Place your CLOCKWORKS card back into the slot.
8. Replace the Apple's cover.
9. Now, turn the power back <ON>.
10. Point the transducer element (looks like a small speaker) to your BSR X-10 COMMAND CONSOLE. For reliable operation, the distance between the BSR command console and the transducer element should not exceed 10 feet.

5.2 USING THE BSR X-10 COMMANDS

Your BSR console has 22 command buttons. Buttons 1 through 16 select devices 1 through 16 while the other 6 buttons select the functions: ON, OFF, BRIGHT, DIM, ALL LIGHTS ON, and ALL OFF. These can be activated manually by pressing the desired buttons on the BSR console itself or via your CLOCKWORKS card by outputting the characters corresponding to these buttons as shown below:

CHARACTER	BUTTON
A	1
B	2
C	3
D	4
E	5
F	6
G	7
H	8
I	9
J	10
K	11

CHARACTER	BUTTON
L	12
M	13
N	14
O	15
P	16
Q	ON
R	OFF
S	BRIGHT
T	DIM
U	ALL LIGHTS ON
V	ALL OFF

For example, to turn <ON> modules 1 and 3 you would do the following:

```
10 PRINT CHR$(4);"PR#4"      :REM CLOCKWORKS IN SLOT 4
20 PRINT "ACQ"               :REM 1 AND 3 <ON>
30 PRINT CHR$(4);"PR#0"      :REM RETURN OUTPUT TO SCREEN
```

For the following examples you'll need one lamp module with its unit code dial set to 1, and one appliance module with its dial set to 3. Make sure that the modules and the command console are all set to the same house code. Plug both modules into AC outlets nearby. Plug an incandescent lamp into the lamp module and a small appliance into module 3. (A transistor radio, portable TV set, or any device that you can easily tell whether it is on or off)

Turn on the lamp and appliance from their power switches and verify your installation by pressing buttons 1, then 3, then ON. The appliance and lamp should have come ON. If this did not happen verify the following:

1. The lamp and appliance are both turned <ON>.

2. The HOUSE and UNIT CODE dials are set correctly.

3. you are pressing the buttons long enough for the modules to respond.

The following simple program allows you to enter BSR commands from the keyboard and send them to your BSR console.

```
100 SLOT=4
110 D$=CHR$(4)
120 TEXT:HOME
130 INPUT"BSR COMMANDS:";CM$
140 IF CM$="" THEN END
150 PRINT D$;"PR#";SLOT
160 PRINT CM$
170 PRINT D$;"PR#0"
180 GOTO 130
```

The above test program is available on your clockworks disk and is named 'BSR.TEST'. Execute the program by typing:

```
RUN BSR.TEST <RETURN>
```

The following line appears:

```
BSR COMMANDS:
```

Now enter the following responses and observe the results.

WHAT YOU TYPE	RESULT
AQ	TURNS ON THE LAMP
AR	TURNS OFF THE LAMP
CQ	TURNS ON THE APPLIANCE
CR	TURNS OFF THE APPLIANCE
U	TURNS ON THE LAMP
V	TURNS OFF THE LAMP
ACQ	TURNS ON THE LAMP AND APPLIANCE
ACR	TURNS OFF THE LAMP AND APPLIANCE

5.3 BRIGHT AND DIM CONTROL

The bright and dim commands can be used with lamp modules and wall switch modules to control the intensity level of incandescent lighting. The following program demonstrates the dim command:

```
10 PRINT CHR$(4);"PR#4" :REM CLOCKWORKS IN SLOT 4
20 PRINT "AQTTTTTTTTTT"
30 PRINT CHR$(4); "PR#0":REM RESTORE OUTPUT TO SCREEN
```

Type in the above program and run.

The "A" selects device 1 (THE LAMP) and the "Q" turns it <ON>.The "T"s dim the light in steps. Now if you change line 20 like this:

```
20 PRINT "ASSSSSSSSSS"
```

and re-run the program, you should see the light brighten up again.

Instead of using multiple S's and T's you can issue a duration code "*" to select the amount of brightening or dimming and have it performed in one
continuous sweep instead of steps. Now change line 20 to the following and re-run.

```
20 PRINT "A*PT"
```

The "*" is recognized by your CLOCKWORKS card to indicate that the next character (IN THIS CASE "P") is a duration code. The duration code can be any character from "A" through "Z". With "A" being the shortest duration code and "Z" the longest. The default duration code is the letter "E" and is used by the firmware unless the duration control feature is used.

The following program shows how you can control the lamp brightness based on time.

```
100 TEXT: HOME
110 D$=CHR$(4)
120 SLOT=4
130 PRINT D$;"PR#";SLOT
140 PRINT D$;"IN#";SLOT
150 INPUT "=";W,MO,DD,YY,HH,MI,SS
160 PRINT D$;"IN#0"
170 PRINT D$;"PR#0"
180 VTAB 10: HTAB 15: PRINT "SECONDS:";SS
190 PRINT D$;"PR#";SLOT
200 IF SS < 30 THEN PRINT "AT"
210 IF SS >= 30 THEN PRINT "AS"
220 PRINT  D$;"PR#0"
230 GO TO 130
```

The above program dims the lamp if the seconds count is less than 30 and brightens it if the seconds count is 30 or more the result is that the lamp dims then brightens once per minute.

CHAPTER VI
FOR ASSEMBLY LANGUAGE PROGRAMMERS

It is easy to use Clockworks from assembly language if you choose to use the onboard firmware. To do so you simply set the mode and then read the clock.

6.1 SETTING THE MODE

To set the mode from assembly language you load the accumulator with the mode selection character and you do a 'JSR' to $CN0B, where N is the slot#. Here's an example using slot #4:

```
LDA #'='              ;SELECTS CLOCKWORKS
JSR $C40B             ;NUMERIC MODE
```

6.2 READING THE CLOCK

To read the clock from assembly language, you set the mode by loading the accumulator with a mode selection character and you execute a JSR $CN0B where N is the slot number, then you do a JSR $CN08. On return the time string in the selected format will be located in the keyboard input buffer (starting at $200) and terminated by a carriage return ($8D). An example for reading the clock located in slot 4 is:

```
        LDA #'='
        JSR $C40B       ;SET MODE TO CLOCKWORKS NUMERIC
        JSR $C408       ;READ CLOCK TO INPUT BUFFER
        RTS             ;OR WHATEVER COMES NEXT
```

If you want to display the string you can use the following subroutine:

```
        LDX #0
NEXT    LDA $200,X
        JSR COUT        ;MONITOR OUTPUT ROUTINE
        CMP #$8D        ;CHECK IF END OF STRING
        BEQ END         ;YES-- EXIT
        INX             ;NO DO NEXT CHARACTER
        BNE NEXT        ;BRANCH ALWAYS
END     RTS
```

6.3 THE MOUNTAIN CLOCK FORMAT

To read the time string in mountain computer's Appleclock format from machine language, you do not print a space to the "MODESET" entry point in the firmware, but rather you have to set the mode directly by storing a ZERO in the scratch pad memory address (MODE) located at $5F8+n where n is the slot number.

This example is for slot 4:

```
LDX #$04
LDA #$00
STA $5F8,X
JSR $C408
```

After executing the above code, the time and date in the Appleclock format will be in the keyboard buffer starting at $200. Remember that the first character at $200 will a quotation mark. The time/date will follow starting at $201 and is terminated with the carriage return character ($8D).

6.4 SETTING THE CLOCK FROM ASSEMBLY LANGUAGE

It is very easy to set the clock from within your assembly language programs due to the SET MODE features built right into Clockworks. Simply output the SET MODE character (+ or !) to the clock via $Cn0B, where n is the slot number. Follow that with the data characters and end it with a carriage return character ($8D or $0D). This feature of Clockworks is extremely useful for dedicated applications of the Apple computer.

The following program sets the clock to TUESDAY APRIL 22, 1986 5:58:24 PM.

```
        LDX #$00          ;INITIALIZE COUNTER
NEXT    LDA STRING,X      ;GET X-TH CHARACTER
        JSR $C40B         ;OUTPUT TO CLOCKWORKS IN SLOT 4
        CMP #$8D          ;CHECK IF END OF STRING
        BEQ END           ;EXIT LOOP IF END OF STRING
        INX               ;INCREMENT COUNTER
        BNE NEXT          ;BRANCH ALWAYS
;
STRING  ASC '+2 04 22 86  17 58 24'
        DFB $8D
;
END     BRK               ;OR WHATEVER COMES NEXT
```

In the above , the spaces in the string are not necessary. They are included in this example simply to make it visually easier for you to analyze the date & time setting.

If you shorten the string's length to the following:

```
STRING    ASC '+2 04 22'
```

Only the date is set. The time and year information is not affected. You can also selectively set any portion by using the SKIPOVER "-" feature. For example changing the string to:

```
STRING    ASC '+- --  --  --  09'
```

Here the hours are changed to 9 AM, while everything else remains intact.

Also, you can set the time very precisely (to the nearest 1/32 of a second) by printing the letter "Z" somewhere in the string. It make good sense however to place it right after the "+" or "!" characters.

6.5 READING THE CLOCK WITHOUT FIRMWARE

For most applications you would want to use the built in firmware to read the clock simply because it is easiest. But if you need to work directly with the hardware registers because you are using interrupts or you want to write a special driver, you can write your program following these steps:

1. Save the status of control register A.
2. Select the address of a clock register.
3. Read the data.
4. If rollover occurred, start over.
5. Repeat steps 2,3, and 4 for all data needed.
6. Restore status of control register A.

The following table shows the address code for any digit.

ADDRESS CODE	NOTATION	ACCESS TO:
$00	S1	UNITS DIGIT OF SECONDS
$01	S10	TENS DIGIT OF SECONDS
$02	MI1	UNITS DIGIT OF MINUTES
$03	MI10	TENS DIGIT OF MINUTES
$04	H1	UNITS DIGIT OF HOURS
$05	H10	TENS DIGIT OF HOURS (NOTE 1)
$06	W	DIGIT FOR DAY OF WEEK
$07	D1	UNITS DIGIT OF DAY OF MONTH
$08	D10	TENS DIGIT OF MONTH (NOTE 2)
$09	MO1	UNITS DIGIT OF MONTH
$0A	MO10	TENS DIGIT OF MONTH
$0B	Y1	UNITS DIGIT OF YEAR
$0C	Y10	TENS DIGIT OF YEAR

* NOTE 1 When reading the tens digit of the hour, only bits 0 and 1 are significant. Bit 3 is high for 24 hour format, low for AM/PM. If in AM/PM format then bit 2 indicates PM when set, and indicates AM when clear.

* NOTE 2 When reading the day of the month Bit 3 and 4 are not significant. Normally these are reset to 0 which indicates that the clock is in the USA/Gregorian method for leap year detection.

There are two other registers in the clock that have specialized use:

1) Reset pre-stages: a write to clock address $D resets 5 DIVIDE-BY-TWO stages before the seconds register providing a precise way to set the time to the nearest 1/32 of a second.

2) Select reference signals: by writing a $E or $F to the address latch, reference signals of 1024 Hz, 1 per second, 1 per minute, and 1 per hour are output from the clock to the PIA handshake lines CA1,CA2,CB1, and CB2 respectively. See the program "TBI" in the section entitled "Reading the time from within an interrupt".

6.6 SENDING BSR COMMANDS FROM ASSEMBLY LANGUAGE

BSR control commands can be sent to your BSR command console by outputting the selected characters to the "WRITE" entry point ($Cn0B) in the clock's firmware. This sample program demonstrates this:

```
            LDX #$00
NEXT        LDA COMMANDS,X
            BEQ END
            JSR $C40B                    ;SLOT 4
            INX
            BNE NEXT
;
END         JMP $FF65                    ;BEEP & EXIT TO MONITOR
;
COMMANDS    ASC 'ABCQ A*WT A*WS'
            DFB $00
```

The BSR string above selects devices 1,2 & 3 and turns them <ON> then sets the duration to "W", dims device 1 (it better be a lamp), sets the duration to "K" and brightens it about halfway.

CHAPTER VII
USING CLOCKWORKS FROM PASCAL

The Pascal support programs are found on the Pascal disk supplied with Clockworks. In the following subchapter you'll find instructions for relocating these programs to your Pascal disks.

7.1 INSTALLING THE CLOCKWORKS UNIT INTO YOUR PASCAL SYSTEM.LIBRARY

The CLOCKWORKS library intrinsic unit gives Pascal programs the capability to use the Clockworks real time clock card. The compiled and linked unit is provided in the file CW:CLOCKWORKS.LIB, ready to install into your own SYSTEM.LIBRARY on your boot disk. If you're not familiar with how to do this, follow these step-by-step directions:

Put the CW: disk in your second drive.

Enter the F(iler by pressing F at the Command prompt.

Transfer CW:CLOCKWORKS.LIB to your boot disk.

Type `TCW:CLOCKWORKS.LIB,*$` then press RETURN.

Press Q to Q(uit from the Filer.

Take the CW: disk out of the second drive, and replace it with a copy of the APPLE2: disk (supplied with your Apple Pascal system).

Execute the Librarian program. Type `XAPPLE2:LIBRARY` then press RETURN.

Type `*LIB.CODE` for the name of the output code file, then press RETURN.

Type `*` for the name of the link code file, then press RETURN.

Type `=` to copy everything from your old SYSTEM.LIBRARY into the new library. Don't press RETURN for this one.

Type `N*CLOCKWORKS.LIB` And press RETURN to select the new link code file.

Type 1 to select CLOCKWORKS and press SPACE. Now type the number of the first empty "slot" in the new library at the bottom of the screen. For example, if 7 is the first place in the new library that doesn't have any unit name after it, type a 7 and press RETURN. This will copy the CLOCKWORKS unit into your new library file.

Press Q to quit, then press RETURN when the librarian says `*Notice?*`.

Press `F` to enter the Filer.

Type `R*CLOCKWORKS.LIB,*SYSTEM.LIBRARY` and press RETURN to remove the old files which have been combined into LIB.CODE. Press Y to update the disk.

Type `C*LIB.CODE,*SYSTEM.LIBRARY` to change LIB.CODE into the new library file.

Type `K*` then press RETURN to K(runch the disk. Press Y to complete the action.

You're now done updating your system library. If you want Clockworks to automatically update your Pascal system date

whenever you start using Pascal, There are two programs on the supplied disk that you may use as a SYSTEM.STARTUP program to accomplish the task.

If you want your Pascal system to start out with a graphic clock face image showing the current Clockworks time, type

`TCW:TICKTOCK.CODE,*SYSTEM.STARTUP` and press RETURN.

If you want something less ornate, and you still want the system date updated to the current Clockworks date, type

`TCW:STARTUP.CODE,*SYSTEM.STARTUP` and press RETURN.

Type Q to quit from the Filer when you're done.

For advanced Pascal programmers:

If you have installed units in your library that use segment number 26, you should edit and recompile CLOCKWORKS.TEXT to change its code segment number in the first line of the program to a number that doesn't conflict with the other units in your library. Be sure to link your recompiled CLOCKWORKS unit with CW.ASM.CODE before installing it in the library. Any programs that use the unit will have to be recompiled after revising the library, so that they will use your revised segment number.

7.2 USING THE CLOCKWORKS UNIT WITH PASCAL

By using the CLOCKWORKS intrinsic unit, your Pascal programs are provided with the DATE, TIMEOFDAY, CLOCKINFO, and SETTIME procedures to read and set the Clockworks real time clock/calendar card.

To use the facilities of the CLOCKWORKS unit, the program must have a USES declaration containing the identifier CLOCKWORKS, immediately after the program heading; for example:

```
PROGRAM TIMELY;
USES APPLESTUFF, CLOCKWORKS;
...
```

For more on units, consult the Pascal programmer's manuals.

The CLOCKWORKS procedures are compatible with their namesakes in the Apple III APPLESTUFF unit. If you produce Pascal programs to run on both the Apple II and the Apple III computers, you can now use the same clock access procedures for both machines by declaring USES CLOCKSTUFF on the Apple II version.

The DATE procedure has the form

DATE(D)

where D is a string variable to contain the information returned by DATE. The returned eight-character string has the format YYYYMMDD, where YYYY is the year (from 1980 to 2079), MM is the month (as a number from 01 to 12), and DD is the day of the month (from 01 to 31). Hence, the date July 4, 2076 would be represented as 20760704. If there is no Clockworks card present, 00000000 is returned.

The TIMEOFDAY procedure has the form

TIMEOFDAY(T)

where T is a string variable to contain the information returned by TIMEOFDAY. The returned six-character string has the format HHMMSS, where HH is the hour in 24-hour format (from 00 to 23), MM is the minute (from 00 to 59), and SS is the second (from 00 to 59). The time 3:35:24 PM would be represented as 153524. If there's no Clockworks card present, 000000 is returned.

The CLOCKINFO procedure has the form

CLOCKINFO(YEAR, MON, DAY, DAYOFWK, HR, MIN, SEC, THOU)

where all of the parameters are integer variables used to contain the date and time information returned by CLOCKINFO. After a CLOCKINFO call, the variables have the following values:

YEAR	year	1980..2079
MON	month	1..12 (Jan..Dec)
DAY	day of the month	1..31
DAYOFWK	day of the week	1..7 (Sun..Sat)
HR	hour	0..23 (24 hour format)
MIN	minute	0..59
SEC	second	0..59
THOU	dummy milliseconds	0

If there is no Clockworks card present, all values are set to zero.

The SETTIME procedure sets the Clockworks time and date. It has the form

SETTIME(T)

where T is an eighteen-character string representing the date and time to be written into the Clockworks card. This has the format

YYYYMMDDWHHNNSSUUU

where the fields are:

YYYY	year	1980..2079
MM	month	01..12 (Jan..Dec)
DD	day of the month	01..31
W	day of the week	1..7 (Sun..Sat)
HH	hour	00..23
NN	minute	00..59
SS	second	00..59
UUU	dummy milliseconds	000..999 (ignored)

The Clockworks card automatically sets its internal milliseconds to zero when SETTIME is called. The UUU part of the format is for compatibility with the Apple III version of SETTIME.
SETTIME doesn't check the information before writing it to the clock.
In order to change the time and/or date settings on the Clockworks card, Switch 2 on the card must be ON.

CHAPTER VIII
ABOUT THE AUXILIARY I/O PORT

Some real time clocks include one or two bits of I/O that are used with an optional module for BSR control. Clockworks has a 16-pin auxiliary connector with a full 8-bit bidirectional port with 2 lines for handshake. Additionally, the connector supplies +5 volts,-5 volts,+12 volts, -12 volts and a 1 MHz clock signal. These signals are available for use by external circuits so long as they are properly designed and do not overload the power supply in your Apple. The applications for this auxiliary connector are limited only by your imagination. Some of the applications include BSR control, direct control of external devices, printer driver, A/D and D/A conversion, plus many more. A BSR interface is available for ClockWorks which is used in conjunction with the BSR X-10 home control system to control lights and appliances from your Apple computer.(See the section entitled "THE ULTRASONIC REMOTE CONTROL OPTION")

8.1 AUXILIARY PORT PINOUTS

The pinout for the auxiliary connector is as follows:

PIN#	FUNCTION	NOTES
1	GND	
2	-12V	
3	+12V	
4	-5V	
5	1 MHz	
6	+5V	
7	CB2	(1)
8	CB1	(2)
9	PB0	
10	PB1	
11	PB2	
12	PB3	
13	PB4	
14	PB5	
15	PB6	
16	PB7	

TOP RIGHT of P.C.B.

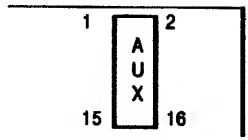

DIAGRAM TO IDENTIFY AUXILIARY CONNECTOR AND PIN-OUT

(1) To enable the CB1 handshake line switch #5 must be opened and switch #8 must be closed. (refer to section 3).
(2) To enable the CB2 handshake line switch #6 must be opened and switch #7 must be closed (refer to section 3).

For detailed information on programming this port please refer to the Motorola MC6821 or the HITACHI HD6821 PIA data sheet.

8.2 EXAMPLE INPUT PROGRAMMING

To demonstrate the use of the input function of the auxiliary port, the following two programming examples are provided here. The first is an Applesoft program that configures the port for input then reads the logic levels at the 8-bit port and displays it on the screen as a number ranging from 0 to 255.

```
100 TEXT:HOME:SLOT=4          :REM CLEAR SCREEN, CLOCK  IN SLOT 4
110 PB= -16254 + SLOT*16      :REM ESTABLISH ADDRESS FOR PORT B
120 CB= PB+1                  :REM CONTROL REGISTER B
130 POKE CB,0                 :REM SELECT DATA DIRECTION REGISTER
140 POKE PB,0                 :REM MAKE ALL LINE INPUTS
150 POKE CB,4                 :REM SELECT PORT FOR ADDRESSING
160 DATA= PEEK(PB)            :REM READ THE DATA AT PORT
170 HTAB1:PRINT SPC(3);       :REM WIPE OLD DISPLAYED VALUE
180 HTAB1:PRINT DATA;         :REM PRINT NEW VALUE
190 GOTO 160                  :REM REPEAT
```

The following is a similar function assembly language program that programs the port for input, and continuously reads the data. It displays the the data in hexadecimal format ranging from $00 to $FF.

```
PRBYTE    EQU  $FDDA      ;PRINTS HEXADECIMAL BYTE IN ACC
COUT      EQU  $FDED      ;OUTPUT ASCII IN "A" TO ACTIVE OUTPUT DEVICE
PORTB     EQU  $C0C2      ;PORTB ADDRESS FOR SLOT 4   "$C0(slot+8)2"
CRB       EQU  PORTB+1    ;CONTROL REGISTER ALWAYS ONE PAST ITS PORT
;
          LDA  #$00       ;SELECT DATA DIRECTION REGISTER
          STA  CRB
          STA  PORTB      ;MAKE ALL LINES INPUT
          LDA  #$04       ;
          STA  CRB        ;SELECT ACCESS TO PORT
;
LOOP      LDA  PORTB      ;READ PORT
          JSR  PRBYTE     ;PRINT VALUE AS TWO HEX DIGITS
          LDA  #$88       ;BACKSPACE CHARACTER
          JSR  COUT       ;BACKSPACE TWICE TO THE BEGINNING
          JSR  COUT
          BNE  LOOP       ;BRANCH ALWAYS AND REPEAT
```

The above programs are intended to describe a simple input use of the auxiliary port. This port can be also be used to receive data from other digital equipment, external keyboard, analog to digital converters, and more. With the above program, an external device could signal the presence of a new data byte by momentarily taking the CB1 input low. This would cause BIT7 of the CRB to be set. A program could easily test this with the BIT instruction. If CRB bit 7 is set, you can read PORTB which clears CRB bit 7.

8.3 EXAMPLE OUTPUT PROGRAMMING

Following the example format of the previous section, the following two programs demonstrate programming the auxiliary port for output. The first example is in Applesoft, it starts by initializing the port for output and then continuously increments the value stored in the port until it reaches 255 (all high), then it recycles.

```
100 TEXT:HOME:SLOT=4                    :REM CLEAR SCREEN, CLOCK  IN SLOT 4
110 PB= -16254 + SLOT*16                :REM ESTABLISH ADDRESS FOR PORT B
120 CB= PB+1                            :REM CONTROL REGISTER B
130 POKE CB,0                           :REM SELECT DATA DIRECTION REGISTER
140 POKE PB,255                         :REM MAKE ALL LINES OUTPUTS
150 POKE CB,4                           :REM SELECT PORT FOR ADDRESSING
160 FOR X=0 TO 255                      :REM LOOP FOR ALL POSSIBLE VALUES
170 POKE PB,X:NEXT                      :REM OUTPUT TO PORT AND REPEAT
180 GOTO 160                            :REM REPEAT FOR NEXT LOOP
```

The assembly language version of this is shown below:

```
WAIT        EQU  $FCA8      ;APPLE'S DELAY ROUTINE
PORTB       EQU  $C0C2      ;PORTB ADDRESS FOR SLOT 4   "$C0(slot+8)2"
CRB         EQU  PORTB+1    ;CONTROL REGISTER ALWAYS ONE PAST ITS PORT
;
            LDA  #$00       ;SELECT DATA DIRECTION REGISTER
            STA  CRB
            LDA  #$FF
            STA  PORTB      ;MAKE ALL LINES OUTPUT
            LDA  #$04       ;
            STA  CRB        ;SELECT ACCESS TO PORT
LOOP        LDA  COUNTER    ;GET COUNT
            STA  PORTB      ;OUTPUT COUNTER VALUE
            LDA  DELAY      ;GET "SLOW DOWN" VALUE
            JSR  WAIT       ;DELAY EXECUTION
            INC  COUNTER    ;ADD ONE TO VALUE TO BE OUTPUT
            JMP  LOOP       ;REPEAT
;
DELAY       DFB  $80
COUNTER     DFB  $00
```

Note that a changeable delay value of 80 was used to allow easy viewability of signal level by using a logic probe. To view and measure higher frequencies you may employ an oscilloscope or a digital frequency counter.

This example is intended for demonstrating the kind of programming used to gain access to the output port function. You would need this information if you intend to use this port for direct control of external devices, to control relays, or to send files/data to another device (such as computers, printers, or digital to analog converters). You can also cause an automatic low level pulse to be generated on handshake output CB2 every time a new value is stored by replacing the "LDA #$04" above with "LDA #$2C", or by using "LDA $24", a low level on CB2 is affected to be driven back high when the external device signals with a low pulse on CB1.

CHAPTER IX
USING INTERRUPTS

Your Clockworks has the ability to generate both IRQ or NMI interrupts in four different frequencies. This allows for minimum interrupt overhead and maximum flexibility.

9.1 WHY USE INTERRUPTS?

Interrupts generated at predetermined intervals add a new dimension to the processing capability of your Apple. Interrupts allow your Apple to do two things at once. For example, as your word processing program is running on your Apple an interrupt occurring once per minute can continuously display the time on the screen without affecting your word processor program. Other applications include gathering data from the auxiliary port at exact intervals and placing it in a buffer to be processed by the "foreground" program running at the same time. As you can see use of interrupts can satisfy many specialized needs that would otherwise require two computers.

9.2 SELECTING THE TYPE AND FREQUENCY OF INTERRUPTS

You can select one of two types of interrupts. The interrupt request (IRQ) and the nonmaskable interrupt (NMI). IRQ type interrupts can be disabled with the 'SEI' "set interrupt disable" instruction, while NMI type interrupts always take priority and cannot be ignored by the processor.

To select IRQ type interrupts, switch #3 must be on (closed) and switch #4 must be off (open). Conversely, to select NMI type interrupts, switch #3 must be off (open) and switch #4 must be on (closed). Non-Maskable interrupts should not be used with a disk operating system or any other programs that depend on timing by software such as communications software that uses timed loops, software that uses the tape or disk I/O, etc... , unless this interrupt is disabled at the source before entering time sensitive code and re-established afterwards. The IRQ can be used in conjunction with most existing software provided that the interrupt disable flag in the microprocessor is set with the 'SEI' instruction before entering time sensitive code and cleared with the 'CLI' instruction immediately afterwards. To select an interrupt frequency you must set the PIA (Peripheral Interface Adapter) control registers as follows:

CONTROL REG. CRA	CONTROL REG. CRB	FREQUENCY	NOTES
04	04	none	
05	04	1024 per second	
0C	04	1 per second	
04	05	1 per minute	(1)
04	0C	1 per hour	(2)

(1) To select interrupts of one per minute switch #5 should be on while switch #8 should be off.
(2) To select interrupts of one per hour switch #6 should be on while switch #7 should be off.

9.3 EXTERNAL SOURCE INTERRUPTING

Clockworks has the ability to generate interrupts when signalled by an external device. The external device can be as simple as a push button switch or as complex as heart rate monitoring equiptment. Two inputs are available for this purpose and are called CB1 and CB2. These can be set up from your software to be active on high-to-low (negative) or on low-to-high (positive) transitions by simply storing the values shown below in CRB, addressed through $C0(n+8)3, where n is the slot number.

SWITCH SETTINGS	CRB	SOURCE	ACTIVE TRANSITION
SW#6 OFF, SW#8 ON	05	CB1	NEGATIVE
	07	CB1	POSITIVE
SW#5 OFF, SW#7 ON	0C	CB2	NEGATIVE
	1C	CB2	POSITIVE

When an interrupt occurs, you can find out where the interrupt came from by examining control register B (CRB). Bit 7 of the CRB will be 'set' if the interrupt came from input CB1 and bit 6 will be set if the cause was CB2.

9.4 PATCHING DOS 3.3 FOR INTERRUPTS

A popular disk operating system for the Apple is DOS 3.3 which is incompatible with interrupts due to a conflict with the firmware built into the Apple ROMS. Both DOS 3.3 and the monitor firmware use a page zero location ($45). If an IRQ interrupt occurs during the execution of DOS routines the monitor changes the value in $45 and DOS may loose a variable. The easiest and most common solution to this problem is to patch DOS to use a different address for variable storage. The following program will modify DOS 3.3 right into memory to make it compatible with interrupts (this program is available on your Clockworks Diskette and is called "DOS 3.3 INTERRUPT PATCH").

```
100 REM PREPARE DOS 3.3 FOR INTERRUPTS
110 READ A : IF A <> 0 THEN POKE A,70: GOTO 110
120 READ A : IF A <> 0 THEN POKE A,44: GOTO 120
1000 DATA 41267,41278,41304,41406,41427,41448,41463,41465,41473,41676,42855,42879,44474,
          44554,44628,44632,48851,48918,48953,48981,48983,48987,49053,49059,49061,0
1010 DATA 47622,48548,0
```

The DOS on the clockworks diskette is already patched. If you want to make interrupt compatible disks all you have to do is boot with the Clockworks DOS 3.3 disk and initialize as many blank disks as you like by using the DOS "INIT" command.

9.5 READING THE TIME FROM WITHIN AN INTERRUPT

When using interrupts to read the clock you should not use the built in firmware for several reasons. One is that the time is read into the keyboard buffer which may contain other vital data. Another is that the Clockworks ROM firmware is mapped into the $C800-$CFFF space and if you have other

cards whose firmware maps into this address space in your Apple then things will go haywire. Another reason is that you'll probably only want to read certain registers from the clock which you can do faster than having to read all the registers, this reduces the interrupt overhead. The following program shows how to set up interrupts of one per second, read the time and display it in the top line of the screen.

```
SOURCE   FILE #01 =>TBI
0000:              1 ;PROGRAM: TIME BY INTERRUPT
0000:              2 ;FUNCTION: DISPLAY CLOCK DATA USING INTERRUPTS
0000:              3 ;
0000:       C080   4 PORTA     EQU   $C080        ;PORT A BASE ADDRESS
0000:       C081   5 CTRLA     EQU   PORTA+1      ;CONTROL REGISTER
0000:              6 ;
0000:              7 ;
-- -- - NEXT OBJECT FILE NAME IS TBI.0
0300:       0300   8           ORG   $300         ;USUALLY UNUSED MEMORY SPACE
0300:              9 ;
0300:78           10 START     SEI                ;DISABLE INTERRUPTS
0301:A9 12        11           LDA   #>INTERRUPT  ;LOAD INTERRUPT VECTOR
0303:8D FE 03     12           STA   $3FE         ;WITH START OF INTERRUPT
0306:A9 03        13           LDA   #<INTERRUPT  ;HANDLER.
0308:8D FF 03     14           STA   $3FF         ;
030B:20 71 03     15           JSR   SEL.FREQ     ;ESTABLISH INTERRUPT RATE
030E:58           16           CLI                ;ENABLE INTERRUPTS
030F:60           17           RTS                ;RETURN
0310:             18 ;
0310:40           19 SLOTNO    DFB   $40          ;FOR CLOCKWORKS IN SLOT 4
0311:       0001  20 COUNTER    DS    1            ;DEFINE STORAGE FOR FORMAT COUNTER
0312:             21 ;
0312:A5 45        22 INTERRUPT LDA   $45          ;TRUE ACCUMULATOR AFTER IRQ
0314:48           23           PHA
0315:8A           24           TXA                ;
0316:48           25           PHA                ;SAVE REGISTERS
0317:98           26           TYA                ;BY PUSHING THEM ON THE STACK.
0318:48           27           PHA
0319:AE 10 03     28           LDX   SLOTNO       ;GET INDEX TO CLOCK'S SLOT
031C:A9 00        29 AGAIN     LDA   #0
031E:8D 11 03     30           STA   COUNTER      ;INITILIZE COUNTER TO ZERO
0321:AC 11 03     31 NEXT      LDY   COUNTER      ;GET FORMAT POSITION COUNT
0324:B9 5F 03     32           LDA   TABLE,Y      ;STEP THRU FORMAT TABLE
0327:10 06   032F 33 B4        BPL   REGCODE      ;BRANCH IF REGISTER SELECT CODE.
0329:C9 FF        34           CMP   #$FF         ;CHECK IF END OF TABLE
032B:F0 29   0356 35           BEQ   EXIT         ;YES.... EXIT
032D:D0 1C   034B 36           BNE   MARK         ;NO... THEN IT MUST BE A SEPARATOR
032F:A8           37 REGCODE   TAY                ;SAVE ADDRESS CODE IN Y
0330:20 82 03     38           JSR   SELECT       ;SELECT CLOCK REGISTER TO READ
0333:BD 80 C0     39           LDA   PORTA,X      ;READ DATA
0336:2C 27 03     40           BIT   B4           ;CLOCK ROLLOVER? (BPL OPCODE = 00010000)
0339:F0 E1   031C 41           BEQ   AGAIN        ;START OVER TO GET A GUARANTEED READING
033B:C0 05        42           CPY   #$05         ;CHECK IF MSB OF HOURS
033D:D0 02   0341 43           BNE   R5
033F:29 03        44           AND   #$03         ;MASK OFF 24 HR AND AM/PM FLAGS
0341:C0 08        45 R5        CPY   #$08         ;CHECK IF MSB OF DAY OF MONTH
0343:D0 02   0347 46           BNE   R6
0345:29 03        47           AND   #$03         ;MASK OFF LEAP YEAR MODE SETTING
0347:29 0F        48 R6        AND   #$0F
0349:09 B0        49           ORA   #$B0         ;MAKE ASCII CODED DIGIT
034B:AC 11 03     50 MARK      LDY   COUNTER      ;PUT COUNTER VALUE IN Y
034E:99 17 04     51           STA   $417,Y       ;STORE DIRECTLY TO SCREEN MEMORY
```

```
0351:EE 11 03       52          INC   COUNTER       ;INCREMENT COUNTER
0354:D0 CB   0321   53          BNE   NEXT          ;BRANCH ALWAYS
0356:               54 ;
0356:20 71 03       55 EXIT     JSR   SEL.FREQ     ;RE-ESTABLISH INTERRUPT RATE
0359:68             56          PLA                 ;RESTORE ALL
035A:A8             57          TAY                 ;REGISTERS
035B:68             58          PLA
035C:AA             59          TAX
035D:68             60          PLA
035E:40             61          RTI                 ;RETURN FROM INTERRUPT
035F:               62 ;
035F:               63 ;THE FOLLOWING TABLE CONTROLS THE FORMAT OF CLOCK DISPLAY
035F:               64 ;CLOCK ADDRESS CODES RANGE FROM $00 TO $0C
035F:               65 ;FORMATTING CODES (/,:,<SPACE>) HAVE MSB SET
035F:               66 ;FORMAT TERMINATOR IS $FF
035F:               67 ;
035F:0A 09 AF       68 TABLE    DFB   $0A,$09,$AF
0362:08 07 AF 0C    69          DFB   $08,$07,$AF,$0C,$0B,$A0,$05,$04
036A:BA 03 02 BA    70          DFB   $BA,$03,$02,$BA,$01,$00,$FF
0371:               71 ;
0371:A9 0F          72 SEL.FREQ LDA   #$0F          ;SET UP INTERRUPT
0373:AE 10 03       73          LDX   SLOTN0        ;LOAD INDEX TO SLOT USED
0376:20 82 03       74          JSR   SELECT
0379:A9 0C          75          LDA   #$0C          ;ENABLE ONCE PER SECOND INTERRUPT
037B:9D 81 C0       76          STA   CTRLA,X
037E:BD 80 C0       77          LDA   PORTA,X       ;CLEAR INTERRUPT STATUS
0381:60             78          RTS
0382:               79 ;
0382:               80 ;LATCH ADDRESS CODE TO CLOCK
0382:               81 ;ON INPUT A=ADDRESS CODE  X=N0 WHERE N IS THE SLOT #
0382:               82 ;ON OUTPUT A IS DESTROYED.  X AND Y UNAFFECTED.
0382:               83 ;
0382:48             84 SELECT   PHA                 ;PUSH A ON STACK
0383:A9 EF          85          LDA   #$EF          ;11101111
0385:20 95 03       86          JSR   DIRECTION     ;ALL OUTPUT EXCEPT BIT 4
0388:68             87          PLA                 ;RESTORE A FROM STACK
0389:09 80          88          ORA   #$80          ;
038B:9D 80 C0       89          STA   PORTA,X       ;WRITE ADDRESS CODE TO CLOCK
038E:49 A0          90          EOR   #$A0
0390:9D 80 C0       91          STA   PORTA,X       ;READ DATA AT ADDRESS SELECTED
0393:               92 ;
0393:A9 E0          93          LDA   #$E0          ;11100000 (FALL INTO DIRECTIOIN ROUTINE)
0395:               94 ;
0395:               95 ;SET PORT DIRECTION
0395:               96 ;ON INPUT, DATA DIRECTION MASK BYTE IN A (BIT SET = OUTPUT), SLOTN0 IN X
0395:               97 ;PRESERVES X AND Y, A IS CLOBBERED
0395:               98 ;
0395:48             99 DIRECTION PHA
0396:A9 00         100          LDA   #0
0398:9D 81 C0      101          STA   CTRLA,X       ;SELECT DIRECTION REGISTER
039B:68            102          PLA                 ;
039C:9D 80 C0      103          STA   PORTA,X       ;SET ACCORDING TO "A"
039F:A9 04         104          LDA   #4            ;
03A1:9D 81 C0      105          STA   CTRLA,X       ;DESELECT DIRECTION REGISTER
03A4:60            106          RTS                 ;RETURN

** SUCCESSFUL ASSEMBLY := NO ERRORS
** ASSEMBLER CREATED ON 15-JAN-84 21:28
** TOTAL LINES ASSEMBLED   106
** FREE SPACE PAGE COUNT   84
```

9.6 THE MILLISECONDS UTILITY

On the CLOCKWORKS DOS 3.3 disk is a utility program that can be use to obtain Milliseconds. The name of the source code file is "MILLISECONDS" and that of the object code is "MILLISECONDS.0". This program bloads at $300 (hexadecimal) or 768 (decimal). Once loaded, the following calls can be executed from your Applesoft program:

CALL 768 Finds CLOCKWORKS and sets location 793 to slotX16. A PRINT PEEK(793)/16 following this CALL will print the slot # of the clock. A zero indicates that Clockworks was not found.

CALL 771 Sets up interrupt vector, resets the counter to zero and enables counting.

CALL 774 Resets the counter to zero.

CALL 777 Copies the contents of the 3 byte counter to decimal memory locations 787 thru 789 where it will not be overwritten.

CALL 780 Stops millisecnds counting.

CALL 783 Starts/continues counting after a CALL 780.

On the same disk, the program "STOPWATCH" can be found. This program uses the Milliseconds utility to maintain a Milliseconds stopwatch on the screen.

```
SOURCE   FILE #01 =>MILLISECONDS
0000:               1 ;PROGRAM: MILLISECONDS
0000:               2 ;FUNCTION: MAINTAIN MILLISECONDS COUNTER IN THREE BYTES
0000:               3 ;
0000:       C080    4 PORTA     EQU   $C080        ;PORT A BASE ADDRESS
0000:       C081    5 CTRLA     EQU   PORTA+1      ;CONTROL REGISTER
0000:               6 ;
0000:               7 ;
-- -- - NEXT OBJECT FILE NAME IS MILLISECONDS.0
0300:       0300    8           ORG   $300         ;USUALLY UNUSED MEMORY SPACE
0300:               9 ;
0300:4C 1A 03      10 ENTRY.1   JMP   FIND.SLOT    ;CALL 768 LOCATES CLOCK SLOT
0303:4C 4B 03      11 ENTRY.2   JMP   INIT.IRQ     ;CALL 771 STARTS MILISECONDS COUNTER
0306:4C 6E 03      12 ENTRY.3   JMP   RESET.CNT    ;CALL 774 RESETS 3 BYTE COUNTER TO ZEROES
0309:4C 7D 03      13 ENTRY.4   JMP   COPY.COUNT   ;CALL 777 COPY COUNTER TO HOLDING AREA
030C:4C 11 03      14 ENTRY.5   JMP   DISABLE      ;CALL 780 STOPS COUNTING
030F:58            15 ENTRY.6   CLI                ;CALL 783 CONTINUES COUNTING
0310:60            16           RTS
0311:              17 ;
0311:78            18 DISABLE   SEI                ;SET INTERRUPT DISABLE FLAG
0312:60            19           RTS
0313:              20 ;
0313:       0003   21 HOLD      DS    3            ;RESERVE THREE BYTES FOR SAFE STORAGE
0316:       0003   22 COUNTER   DS    3            ;RESERVE THREE BYTES FOR COUNTING
```

```
0319:          0001   23 SLOTN0     DS    1              ;ONE BYTE FOR SLOT # IN HIGH NIBBLE
031A:                 24 ;
031A:A2 C7            25 FIND.SLOT  LDX   #$C7           ;START WITH SLOT 7
031C:8E 49 03         26 NEXT       STX   CHK.ROM+2      ;SELF MODIFYING SEARCH
031F:A0 00            27            LDY   #0
0321:20 47 03         28            JSR   CHK.ROM
0324:C9 08            29            CMP   #8             ;LOOK FOR 'PHP' AT $CN00
0326:D0 11    0339    30            BNE   NOMATCH
0328:A0 FC            31            LDY   #252
032A:20 47 03         32            JSR   CHK.ROM
032D:C9 C3            33            CMP   #195           ;LOOK FOR HI ASCII 'C' AT $CNFC
032F:D0 08    0339    34            BNE   NOMATCH
0331:C8               35            INY
0332:20 47 03         36            JSR   CHK.ROM
0335:C9 D7            37            CMP   #215           ;LOOK FOR HI ASCII 'W' AT $CNFD
0337:F0 05    033E    38            BEQ   FOUNDIT
0339:CA               39 NOMATCH    DEX
033A:E0 C0            40            CPX   #$C0           ;ALL SLOTS CHECKED?
033C:D0 DE    031C    41            BNE   NEXT           ;NO, DO NEXT SLOT
033E:                 42 ;
033E:8A               43 FOUNDIT    TXA
033F:0A               44            ASL   A
0340:0A               45            ASL   A
0341:0A               46            ASL   A
0342:0A               47            ASL   A
0343:8D 19 03         48            STA   SLOTN0         ;PUT SLOT# X16 AT 'SLOTN0'
0346:60               49            RTS
0347:                 50 ;
0347:B9 00 C7         51 CHK.ROM    LDA   $C700,Y
034A:60               52            RTS
034B:                 53 ;
034B:78               54 INIT.IRQ   SEI                  ;DISABLE INTERRUPTS
034C:A9 8C            55            LDA   #>INTERRUPT    ;LOAD INTERRUPT VECTOR
034E:8D FE 03         56            STA   $3FE           ;WITH START OF INTERRUPT
0351:A9 03            57            LDA   #<INTERRUPT    ;HANDLER.
0353:8D FF 03         58            STA   $3FF           ;
0356:A9 0F            59            LDA   #$0F
0358:AE 19 03         60            LDX   SLOTN0
035B:20 A3 03         61            JSR   SELECT         ;ENABLE REFERENCE FREQUENCIES
035E:A9 05            62            LDA   #$05           ;FOR MILLISECONDS
0360:9D 81 C0         63            STA   CTRLA,X
0363:BD 80 C0         64            LDA   PORTA,X        ;CLEAR PIA STATUS
0366:20 6E 03         65            JSR   RESET.CNT      ;RESET COUNTER TO ZERO
0369:20 7D 03         66            JSR   COPY.COUNT     ;RESET HOLD COUNTER
036C:58               67            CLI                  ;START COUNTING
036D:60               68            RTS                  ;RETURN
036E:                 69 ;
036E:08               70 RESET.CNT  PHP                  ;SAVE PROCESSOR STATUS
036F:78               71            SEI                  ;DISABLE INTERRUPS
0370:A9 00            72            LDA   #0
0372:8D 16 03         73            STA   COUNTER
0375:8D 17 03         74            STA   COUNTER+1      ;RESET ALL THREE BYTES TO ZERO
0378:8D 18 03         75            STA   COUNTER+2
037B:28               76            PLP                  ;RESTORE PROCESSOR STATUS
037C:60               77            RTS                  ;RETURN
037D:                 78 ;
037D:08               79 COPY.COUNT PHP                  ;SAVE PROCESSOR STATUS
037E:78               80            SEI                  ;DISABLE INTERRUPTS
037F:A2 02            81            LDX   #2             ;3 BYTES TO COPY
0381:BD 16 03         82 ALL3       LDA   COUNTER,X      ;GET COUNTER BYTE
0384:9D 13 03         83            STA   HOLD,X         ;PUT IN SAFE PLACE
```

```
0387:CA            84           DEX                ;NEXT BYTE
0388:10 F7  0381   85           BPL   ALL3         ;
038A:28            86           PLP                ;RESTORE PROCESSOR STATUS
038B:60            87           RTS                ;RETURN
038C:              88 ;
038C:8A            89 INTERRUPT TXA                ;ACC ALREADY SAVE D IN $45
038D:48            90           PHA                ;SAVE X BY PUSHING IT ON STACK
038E:AE 19 03      91           LDX   SLOTN0       ;GET INDEX TO SLOT
0391:BD 80 C0      92           LDA   PORTA,X      ;RESET INTERRUPT FLAG
0394:A2 02         93           LDX   #2           ;
0396:FE 16 03      94 INCREMENT INC   COUNTER,X    ;INCREMENT 3 BYTE COUNTER
0399:D0 03  039E   95           BNE   EXIT.IRQ
039B:CA            96           DEX
039C:10 F8  0396   97           BPL   INCREMENT
039E:68            98 EXIT.IRQ  PLA                ;RESTORE X REGISTER
039F:AA            99           TAX
03A0:A5 45        100           LDA   $45          ;RESTORE ACCUMILATOR
03A2:40           101           RTI                ;RETURN FROM INTERRUPT
03A3:             102 ;
03A3:             103 ;LATCH ADDRESS CODE TO CLOCK
03A3:             104 ;ON INPUT A=ADDRESS CODE  X=N0 WHERE N IS THE SLOT #
03A3:             105 ;ON OUTPUT A IS DESTROYED.  X AND Y UNAFFECTED.
03A3:             106 ;
03A3:48           107 SELECT    PHA                ;PUSH A ON STACK
03A4:A9 EF        108           LDA   #$EF         ;11101111
03A6:20 B6 03     109           JSR   DIRECTION    ;ALL OUTPUT EXCEPT BIT 4
03A9:68           110           PLA                ;RESTORE A FROM STACK
03AA:09 80        111           ORA   #$80         ;
03AC:9D 80 C0     112           STA   PORTA,X      ;WRITE ADDRESS CODE TO CLOCK
03AF:49 A0        113           EOR   #$A0
03B1:9D 80 C0     114           STA   PORTA,X      ;READ DATA AT ADDRESS SELECTED
03B4:             115 ;
03B4:A9 E0        116           LDA   #$E0         ;11100000 (FALL INTO DIRECTIOIN ROUTINE)
03B6:             117 ;
03B6:             118 ;SET PORT DIRECTION
03B6:             119 ;ON INPUT, DATA DIRECTION MASK BYTE IN A (BIT SET = OUTPUT), SLOTN0 IN X
03B6:             120 ;PRESERVES X AND Y, A IS CLOBBERED
03B6:             121 ;
03B6:48           122 DIRECTION PHA
03B7:A9 00        123           LDA   #0
03B9:9D 81 C0     124           STA   CTRLA,X      ;SELECT DIRECTION REGISTER
03BC:68           125           PLA                ;
03BD:9D 80 C0     126           STA   PORTA,X      ;SET ACCORDING TO "A"
03C0:A9 04        127           LDA   #4           ;
03C2:9D 81 C0     128           STA   CTRLA,X      ;DESELECT DIRECTION REGISTER
03C5:60           129           RTS                ;RETURN

** SUCCESSFUL ASSEMBLY := NO ERRORS
** ASSEMBLER CREATED ON 15-JAN-84 21:28
** TOTAL LINES ASSEMBLED   129
** FREE SPACE PAGE COUNT   83
```

APPENDIX A
ADJUSTING THE TIME BASE FREQUENCY

The accuracy of the time keeping performed by Clockworks is dependant on the time base frequency generated by the small quartz crystal labled 'XTAL'. The crystal oscillates at 32768 Hz. A fine tuning capacitor (located on the right side of the crystal) was percisely calibrated at the factory. The accuracy is also somewhat affected by the temperature in your Apple. If you notice that the clock is too fast or too slow you may wish to adjust the tuning capacitor by turning the screw a very small amount counter clockwise to slow the clock down or clockwise to speed it up.

APPENDIX B
TESTING THE BATTERY

The lithium coincell battery on your Clockworks is installed in a high quality battery holder for easy replacement, under normal use you will not need to change this battery for many years. But if you notice that the time is erroneous after you turn on the Apple try setting the time, if this does not help or if the problem recurs you'll need to check the battery voltage.

To do this you will need a Voltmeter, preferably the digital type. Turn off your Apple and remove your Clockworks card (HANDLE WITH CARE). Make sure your voltmeter is set up to measure volts. On the circuit side of the card right behind the battery there are two terminals place the tip of one probe of your vlotmeter on one terminal and the other tip on the other terminal.

If the battery reads below 2.6 volts then you should replace it with a new one. These batteries are available from several electronics supply houses or you can order one directly from us.

APPENDIX C
TECHNICAL SPECIFICATIONS & FEATURES

* High quality quartz crystal time base.

* 24 hour military or 12 hour AM/PM formats.

* High capacity lithium battery backup.

* Fully ProDOS and DOS 3.3 compatible.

* Four interrupt frequencies 1024 Hz, 1 per second, 1 per minute, and 1 per hour.

* Two levels of Interrupts "IRQ" and "NMI".

* Supplies the date with month, day of month, year, the day of the week, the hours, minutes and seconds.

* An ideal substitute for any other clock because of the eight built-in formats.

* Super smart firmware in 4096 bytes of EPROM.

* Two set time modes allow setting the clock with a simple print command.

* Built in BSR control command set.

* Automatically detects and corrects for leap years.

* On-board bidirectional 8-bit port with 2 handshake lines.

* Auxiliary connector with +5,+12,-5,and -12 volts and a 1MHz clock reference signals.

* Auxiliary connector can be used to drive BSR control, direct control, printer driver, A/D and D/A conversions, and more....

APPENDIX D
QUICK REFERENCE SUMMARY

	READ MODES:	LANGUAGE
" "	06/24 14:35:45.185	APPLESOFT BASIC
$	MON JUN 24 02:35:45 PM	APPLESOFT BASIC
&	MON JUN 24 14:35:45	APPLESOFT BASIC
#	06,01,24,14,35,45	APPLESOFT BASIC
>	MON JUN 24 02:35:45 PM	INTEGER BASIC
<	MON JUN 24 14:35:45	INTEGER BASIC
:	1 06/24/85 14:35:45	APPLESOFT BASIC
=	01,06,24,85,14,35,45	APPLESOFT BASIC

80 COLUMN COMMAND FORMAT:

```
CALL 49216 + 256 * SLOT,FM$,TM$
```

Where FM$ is a one character string indicating the read format, and TM$ is the variable name in which the clock data will be placed.

SET TIME COMMAND FORMATS:

CLOCKWORKS SET MODE: +W MO DD YY HH MI SS <RETURN>
THUDERCLOCK SET MODE: !MO W DD HH MI SS <RETURN>

The letter 'Z' may be included in the string to zero out fractions of seconds to precisely set the time to better than 1/32 of a second accuracy.
Placement of the letter 'Z' at the begining of strings that set the second is recommended.

BSR CONTROL COMMANDS:

CHARACTER	BUTTON	CHARACTER	BUTTON
A	1	L	12
B	2	M	13
C	3	N	14
D	4	O	15
E	5	P	16
F	6	Q	ON
G	7	R	OFF
H	8	S	BRIGHT
I	9	T	DIM
J	10	U	ALL LIGHTS ON
K	11	V	ALL OFF

The "*" character, when ouput to the clock, indicates that the next character to be a duration code. This can be any letter from "A" (minimum) through "Z" (maximum).